

Testing? Check It Out!

Arkady Galyash

TosChart

May 26, 2014



Agenda

1. How regular JUnit test looks like (in time)?
2. Property based testing
3. Java Quickchecks
4. Where can we use it?
5. Links



Intro

- John Doe The Programmer
- Java developer @ Moon Ms
- Since 2000
- Binary search algorithm

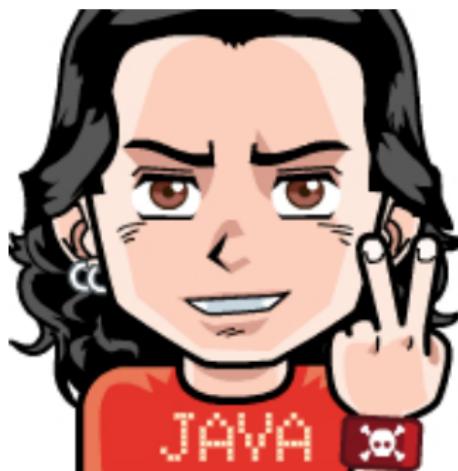


Testing? Check It Out!

└ How regular JUnit test looks like (in time)?

└ June 2000

June 2000



Just graduated, "tests are for chicken"



Testing? Check It Out!

└ How regular JUnit test looks like (in time)?

└ May 2001

May 2001



JUnit discovered



Testing? Check It Out!

└ How regular JUnit test looks like (in time)?

└ July 2006

July 2006



User tries to search non-existent element



Testing? Check It Out!

└ How regular JUnit test looks like (in time)?

└ December 2007

December 2007



User tries to search in array with duplicates



Testing? Check It Out!

└ How regular JUnit test looks like (in time)?

└ October 2009

October 2009



Integer overflow



DEVEXPERTS

Long way



2000

2001

2006

2007

2009

simpleTest



missingTest



sameTest



hugeArrayTest



xUnit

- Design tests example by example
- Test suites give us confidence that code works for the examples we thought of
- If we discover a test case, this test is useless right now, it may be useful only in future
- "Don't ask, don't tell"



Property based testing

- Another approach
- You specify actions and post-conditions
- Let computer generate input data
(it will not forget about corner cases)



QuickCheck History

- Initially written for Haskell in 1999
- By Koen Claessen and John Hughes
- BSD-style License

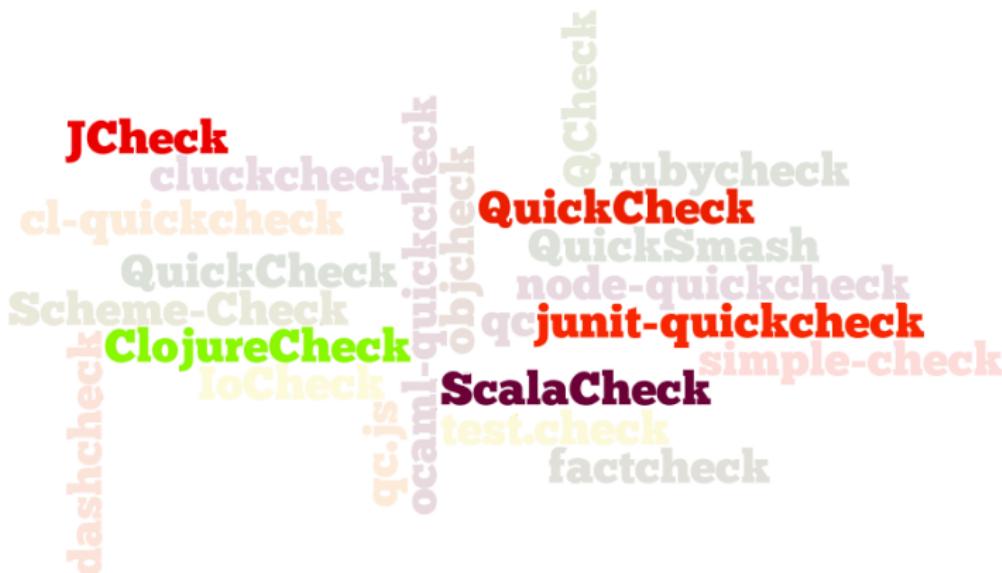
QuickCheck implementations

A word cloud visualization showing various QuickCheck implementations across different programming languages. The words are arranged in a cluster, with larger and more prominent words indicating more popular or well-known implementations. The colors of the words vary, creating a visual hierarchy.

The words visible in the cloud include:

- JCheck
- cluckcheck
- cl-quickcheck
- QuickCheck
- Scheme-Check
- ClojureCheck
- IoCheck
- qc.js
- ocaml-quickcheck
- objccheck
- QuickSmash
- node-quickcheck
- qcjunit-quickcheck
- simple-check
- ScalaCheck
- test.check
- factcheck
- QCheck
- rubycheck

QuickCheck implementations



Scala? Clojure?

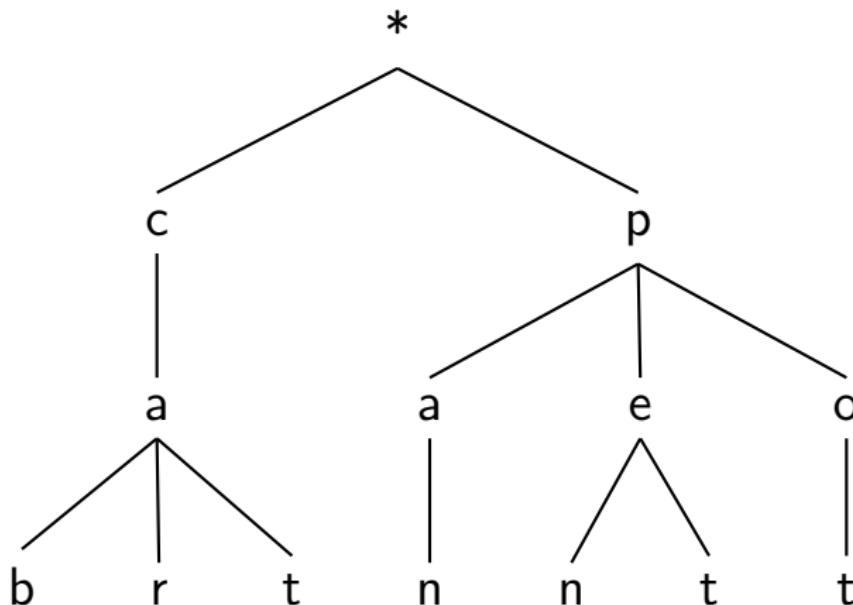


Implementations criteria

- Generators of custom types
- Generators with restrictions
- Search-space customization



Trie

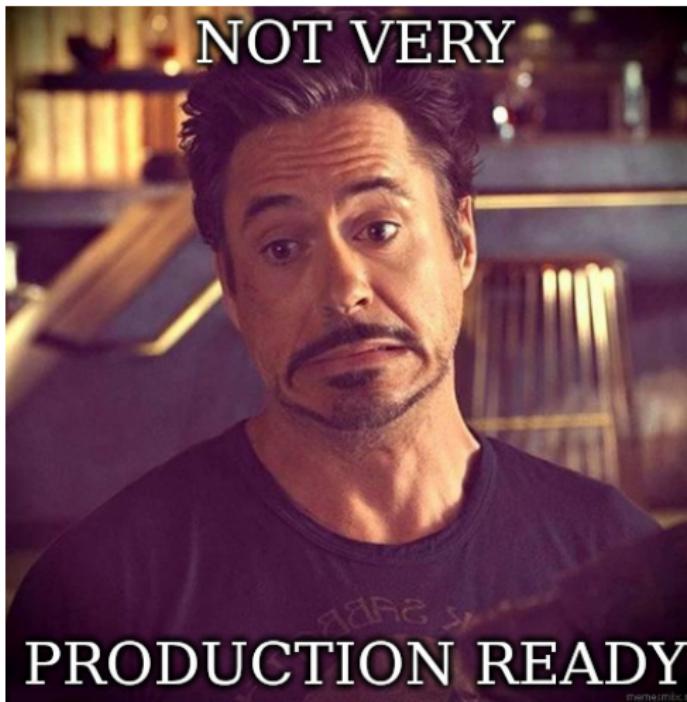


junit-quickcheck

- Based on
`org.junit.experimental.theories.*`
- Annotation-based
- On github.com by @pholser
- 0.3 version in maven repository
- Found [bug with generics](#) during making presentation



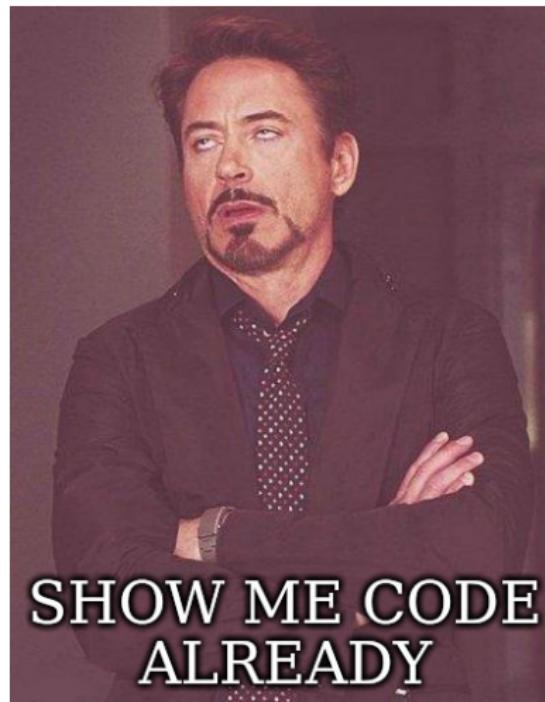
Demo



JCheck

- Has its own JUnit Runner
- Annotation-based
- On [sf.net](#) by @hampusr
- 0.1 version jar from 2008

Demo



quickcheck

- Do not have Runner at all
- Handle only data generation
- On [@blob79](https://bitbucket.org)
- 0.6 version in maven repository



Demo



ThinkScript

- Programming language for traders
- Technical analysis
- Has a lot of built-in functions (SMA, EMA, WMA, ...)



Standard deviation

$$\sigma = \sqrt{\frac{1}{N} \sum_{i=1}^N (x_i - \mu)^2},$$
$$\mu = \frac{1}{N} \sum_{i=1}^N x_i$$

Demo



Donald Knuth is shocked by stdev



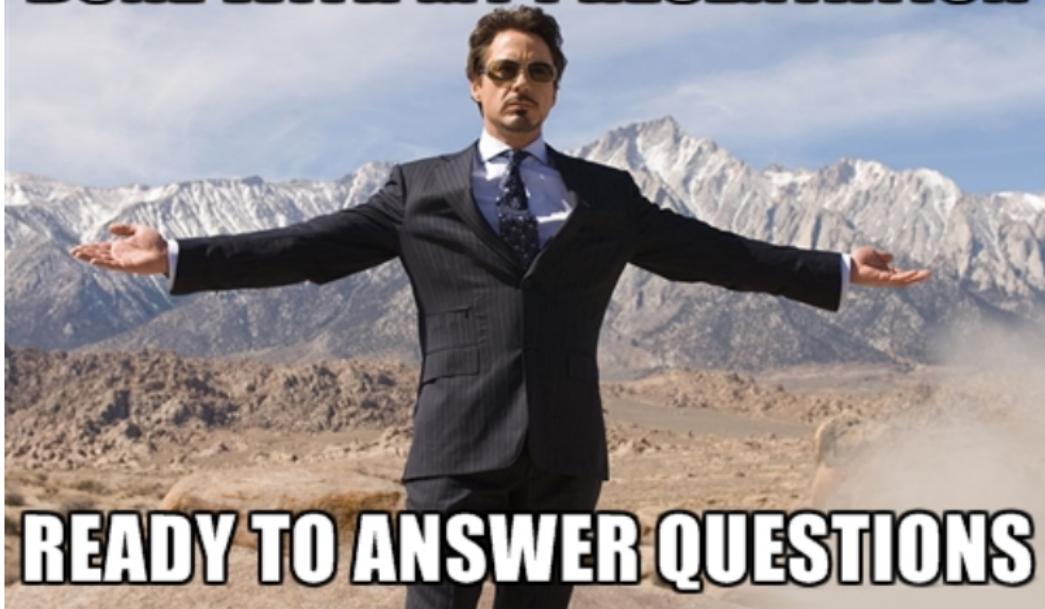
Links

- JohnDoeProject with examples
- Introduction to QuickCheck2
- Blog dedicated to Java Quickcheck
- Better Than Unit Tests



Thank you!

DONE WITH MY PRESENTATION



READY TO ANSWER QUESTIONS